testing the set of control bits of the compressed instruction to determine a condition is identified in which subinstruction sharing is to occur for the compressed instruction;

when subinstruction sharing is determined to occur, parsing the compressed instruction to route the redundant subinstruction to a plurality of functional processing units as determined by the identified condition;

concurrently executing the subinstruction at said plurality of functional processing units. --

---

REMARKS

This amendment is responsive to the Office action mailed November 18, 2002 for the above-captioned application.

- The examiner has objected to the declaration. A substitute declaration is enclosed.
- The examiner has objected to the specification as lacking a description of Fig. 8. The specification has been amended at page 16, line28, inserting the text: "Fig. 8 shows a set 36 of control bit groups 38. Each group 38 includes a plurality of bits 40." A substitute page 16 is attached.
- Claims 2, 5 and 6 have been objected to for informalities. Such claims have been amended as per the examiner's suggestions.
- Claims 13-16, 19 and 20 have been rejected under 35 USC 102(e).
- Claims 1-12, 17, 18 and 21 have been rejected under 35 USC 103(a).

Claims 1, 4, 5 and 6 have been amended. Claims 7-12 have been canceled. No new claims are added. Claims 1-6 and 13-21 remain pending. The rejections to claims 13-21 are respectfully traversed. Reconsideration is requested.

Prior Art Rejections

OA ¶8-16: Claims 13-16, 19 and 20

The examiner has rejected claims 13-16, 19 and 20 under 35 USC 102(e) as being anticipated by U.S. Patent No. 6,173,389 (Pechanek et al.) Pechanek discloses a method and apparatus for selecting subinstructions for execution time parallelism. A given very long word instruction (VLIW) is stored in a corresponding PE's VLIW memory (called a VIM). As shown in Fig. 3 therein, a system includes multiple PE's (PE0-PE3). Each PE includes multiple execution units 131. Note that A VLIW is loaded into the VIM of a given PE. There is no disclosure or suggestion of allocating subinstructions of a common VLIW into execution units across multiple PE's. Accordingly, there is no disclosure or suggestion of loading a subinstruction of a given VLIW into functional units of multiple clusters

Pechanek also discloses an XV1 VLIW instruction, an LV VLIW instruction, and an XV2 VLIW instruction. The LV instruction is to load a VLIW from a PE's VIM. The subinstructions within such VLIW then are allocated among execution units of the PE according to a control instruction XV1 which follows. An XV2 VLIW instruction masks fields of the loaded VLIW instruction so that specific execution units do not operate for a given cycle. Of significance here, is that there are no control bits in the loaded VLIW instruction which determine how the subinstructions of such loaded VLIW instruction are to be allocated among execution units. Such allocation is determined by a separate, subsequent VLIW instruction.

As decribed in Pechanek's title, Pechanek et al. use an instruction to select subinstructions to be active for a given cycle. Pechanek do not disclose or suggest compressing subinstructions.

Claim 13 recites a computer system including a processor and an instruction cache. The processor includes a plurality of clusters of functional processing units, each cluster has a common number of functional processing units. Each instruction executed by the processor comprises ... subinstruction ..., along with a set of control bits. The subinstruction is to be shared by a plurality of functional processing units as determined by the condition of the control bits. Thus, the control bits of a given instruction determine the sharing of the subinstruction included within the given instruction. In contrast, Pechanek uses a separate instruction to determine routing of a subinstruction. Accordingly, claim 13 distinguishes over the cited art based at least upon the claim limitation

- a shared subinstruction stored in a given field of the first instruction which is to be shared by a plurality of the functional processing units, said plurality of functional processing units being determined by said condition of the set of control bits.

Claim 14 depends from claim 13 and distinguishes over the cited art for the same reasons as given for claim 13. Claim 14 further distinguishes over the cited art, based upon the following claim limitation:

- in which said shared subinstruction is for a first functional processing unit of a first cluster and a first functional processing unit of a second cluster when the set of control bits identifies a first prescribed condition.

Note that the subinstruction is detined for functional processing units in different clusters. In Pechanek a given subinstruction of a loaded VLIW instruction is only routed to execution units within the same PE.

Claim 15 depends from claim 14 and distinguishes over the cited art based upon the same reasons as given for claims 13 and 14. Claim 15 further distinguishes over the cited art by reciting that a second subinstruction is shared among functional processing units of a first cluster and a second cluster.

Claim 16 depends from claim 14 and distinguishes over the cited art based at least upon the same reasons as given for claims 13 and 14. Claim 16 further distinguishes over the cited art by including means for routing the subinstruction to ... the first cluster and ... the second cluster.

Claims 19 and 20 depend from claim 14 and distinguish over the cited art based at least upon the same reasons as given for claims 13 and 14.

OA ¶ 17-26: Claims 1-3 and 17

Claims 1-3 and 17 have been rejected under 35 USC 103(a) as being unpatentable over Pechanek et al., in view of the publication of Rosenberg.

Claim 1 recties a method for sharing subinstructions of a given instruction among functional processing units of multiple clusters on a VLIW processor. When the control bits of the given instruction identify a prescribed condition a subinstruction is routed to multiple functional processing units as determine by the prescribed condition. As previoulsy distinguished regarding claim 13, Pechanek does nor use the control bits of a given instruction

to route the subinstructions of such instruction. Pechanek uses a subsequent VLIW instruction to route previously loaded subinstructions.

Further, the examiner asserts that parallel processing meas instructions are executed concurrently in different processing units of different clusters. However, the examiner stretches the definition by asserting that parallel processing means a shared subinstruction routed to different clusters is going to be processed concurrently in the respective clusters. Parallel processing means there is concurrent processing. It does not by definition mean that pipelines are synchronized to assure that the same subinstruction routed to two separate clusters is processed at the same time - rather than 1 or more clock cycles before or earlier.

Claims 2 and 3 depend from claim 1 and distinguish over the cited art for the same reasons as given for claim 1. Claims 2 and 3 further distinguish over the cited art based on the same reasons given in regard to claim 14 above.

Claim 17 depends from claim 14 and distinguishes over the cited art for the same reasons as given for claims 13 and 14. Claim 17 further distinguishes over the cited art for the same reasons as given for claim 1 above.

OA ¶27-32: Claims 4-5

Claims 4-5 have been rejected under 35 USC 103(a) as being unpatentable over Pechanek in view of U.S. Patent No. 6,044,450 (Tsushima). Claim 4 is in independent format and distinguishes over the cited art based upon steps performed during compilation of the computer program. Neither Pechanek nor Tsushima disclose:

- during compilation of the computer program, the steps of:
- identifying a pattern in which a subinstruction occurs more than once in a given instruction...;
- determining whether the pattern is among a setoff prescribed patterns; and
- ... setting a set of control bits for the instruction to indicate that the pattern is present.

Claim 5 distinguishes over the cited art for the same reasons as given for claim 4, and further based on at least the following limitations:

- during compilation of the computer program, compressing the given instruction when the pattern ... by deleting one occurrence of the redundant subinstruction ...

OA ¶33-34: Claim 6

Claim 6 is in independent format and has been rejected under 35 USC 103(a) as being unpatentable over Pechanek in view of Tsushima and further in view of Rosenberg. Claim 6 distinguishes over the cited art for the same reasons as given above for claim 1.

Claims 7-12, 18 and 21.

Claims 7-12 are cancelled. Claims 18 and 21 depend from claim 14 and distinguish over the cited art based at east on the same reasons as given for claims 13 and 14.

Conclusion

In view of the above remarks regarding the cited art, it is respectfully submitted that the claims contain key limitations that are not present in the cited art and not obvious from the cited art. These particular limitations, are not disclosed in or suggested by cited references. These limitations are significant advances over the prior art and resulted in a novel method and apparatus for sharing VLIW subinstructions.

] In view of the above amendments and remarks, it is respectfully submitted that the claims are now in condition for allowance. The Examiner's action to that end is respectfully requested. Reconsideration of the claims and withdrawal of the rejections is respectfully requested.

If, in the opinion of the Examiner, a telephone conference would expedite the prosecution of the application, the Examiner is invited to call the undersigned attorney at the telephone number given below.

Respectfully submitted,

Dated: 2/28/03 By: S. P. Koda

Steven P. Koda
Reg. No. 32,252
Tel.: 206-284-3692

Koda Law Office
P.O. Box 10057
Bainbridge Island, WA. 98110

APPENDIX A - Marked-Up Version of the Claims

The following amendments have been made to claims 1, 4, 5 and 6.

-- 1. (Amended) A method for sharing a subinstruction of a given instruction among functional processing units of a plurality of clusters on a processor having a very long instruction word architecture, the given instruction including a set of control bits and at least one subinstruction, the processor comprising the plurality of clusters, each one cluster of the plurality of clusters comprising a plurality of functional processing units, the method comprising the steps of:

testing the set of control bits of the given instruction to identify a prescribed condition;

when the prescribed condition is identified, routing said subinstruction of the given instruction to multiple functional processing units as determined by the prescribed condition;

concurrently executing the subinstruction at said multiple functional processing units.

2. (Amended) The method of claim 1, in which the step of routing comprises routing said subinstruction of the given instruction to a first functional processing unit of a first cluster of the plurality of clusters and to a first functional processing unit of a second cluster of the plurality of clusters; and in which the step of executing comprises concurrently executing the subinstruction at said first functional processing unit of the first cluster of the plurality of clusters and [to] at the first functional processing unit of the second cluster of the plurality of clusters.

4. (Amended) A method for storing an instruction of a computer program to be executed on a processor having a very long instruction word architecture,

wherein each instruction comprises at least one subinstruction and up to a first prescribed number of subinstructions, the first prescribed number being at least two,

wherein the processor is organized into a plurality of clusters equaling a second prescribed number, each one cluster of the plurality of clusters comprising a common number of functional processing units, wherein the common number of functional processing units times the second prescribed number equals the first prescribed number,

wherein for a given instruction having the first prescribed number of subinstructions, each functional processing unit of the plurality of clusters is for executing a respective subinstruction of the given instruction, the method comprising, during compilation of the computer program, the steps of:

identifying a pattern in which a subinstruction occurs more than once in the given instruction, said subinstruction being a redundant subinstruction;

determining whether the pattern is among a set of prescribed patterns;

when the pattern is among the set of prescribed patterns, setting a set of control bits for the instruction to indicate that said pattern is present.

5. (Amended) The method of claim [3,] 4, further comprising, during compilation of the computer program, compressing the given instruction when the pattern is among the set of prescribed patterns by deleting one occurrence of the redundant subinstruction in the given instruction to achieve a compressed instruction.

6. (Amended) The method of claim 5, further comprising, during run time of the computer program, the steps of:

moving the compressed instruction into an instruction cache;

testing the set of control bits of the compressed instruction to determine a condition is identified in which subinstruction sharing is to occur for the compressed instruction;

when subinstruction sharing is determined to occur, parsing the compressed instruction to route the redundant subinstruction to a plurality of functional processing units as determined by the identified condition;

concurrently executing the subinstruction at said plurality of functional processing units.